```h
#pragma once
#include <Windows.h>

class AudioManager
{
    static AudioManager* s_pInstance;

    bool m_SoundOn;

    AudioManager()
        : m_SoundOn(true)
    {

    }

    //is this a singleton design pattern? one and only one instance?
    // global access, no ownership, lazyinitialisation
    // saves memory - but how?
    // Flexibility

public:
    static AudioManager* GetInstance()
    {
        if (s_pInstance == nullptr)
        {
            s_pInstance = new AudioManager();
        }
        return s_pInstance;
    }

    static void destroyInstance()
    {
        delete s_pInstance;
        s_pInstance = nullptr;
    }

    void ToggleSound()
    {
        m_SoundOn = !m_SoundOn;
    }

    bool IsSoundOn()
    {
        return m_SoundOn;
    }

    void playdoorclose()
    {
        if (!m_SoundOn)
        {
            return;
        }
        Beep(500, 75); // frequency and duration
```

```
54          Beep(500, 75);
55      }
56
57      void playerdooropen()
58      {
59          if (!m_SoundOn)
60          {
61              return;
62          }
63          Beep(1397, 97);
64      }
65
66      void pickupkey()
67      {
68          if (!m_SoundOn)
69          {
70              return;
71          }
72          Beep(1568, 100);
73      }
74
75      void dropKeySound()
76      {
77          if (!m_SoundOn)
78          {
79              return;
80          }
81          Beep(1568, 200);
82          Beep(1568, 50);
83      }
84
85      void moneySound()
86      {
87          if (!m_SoundOn)
88          {
89              return;
90          }
91          Beep(1568, 50);
92      }
93
94      void loseLife()
95      {
96          if (!m_SoundOn)
97          {
98              return;
99          }
100         Beep(200, 100);
101     }
102
103     void PlayLoseSound()
104     {
105         if (!m_SoundOn)
106         {
```

```
107             return;
108         }
109         Beep(500, 75);
110         Beep(500, 75);
111         Beep(500, 75);
112         Beep(500, 75);
113         Beep(500, 75);
114         Beep(500, 75);
115     }
116
117     void win()
118     {
119         if (!m_SoundOn)
120         {
121             return;
122         }
123         Beep(1568, 200);
124         Beep(1568, 200);
125         Beep(1568, 200);
126         Beep(1245, 1000);
127         Beep(1397, 200);
128         Beep(1397, 200);
129         Beep(1397, 200);
130         Beep(1175, 1000);
131     }
132 };
```